Some Results on Qudit Quantum Error-Correction

by

Lane G. Gunderman

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Masters of Science
in
Physics (Quantum Computing)

Waterloo, Ontario, Canada, 2019

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Although throughout I discussed many concepts with peers, I am the sole author of all material contained within chapters 1,2, and 4.

For chapter 3, Andrew Jena and I discussed the work regularly, however, I am the sole author of all material aside from Theorem 16 and Corollary 18. The final write-ups for these results were performed primarily by Andrew Jena. This is indicated with a footnote as well to clarify these contributions.

Abstract

Quantum computing's seemingly perpetual promise of nearness still has a few hurdles to surmount before it can become a reality. Among these hurdles is that of protection of information from random errors. A potential solution for this challenge is stabilizer codes, which are the analog of classical linear error-correcting codes, however, with an additional axis of error possibility. By and large, quantum computing is discussed in the standard qubit, or two-level, language, however, it is worth considering the case of qudits, or more than two-level systems. Often times results follow simply from some form of algebraic extension: typically group theory or linear algebra. In this work we consider some features that are not immediately apparent from that approach, more often appealing to physical intuition to guide our mathematical ideas, then proving these ideas using the language of qudit operators.

Here we consider two particular previously unexplored ideas. The first idea is that of embedding and inscribing of codes into spaces of different sizes than the stabilizer code was originally designed for. Here, we show that all codes can be embedded, and that for infinitely many primes we can in fact guarantee that the distance is at least preserved–a somewhat surprising result. The second idea is, in a way an application of those presented in the first idea, that of turning many stabilizer codes into hybrid codes by taking advantage of relabelling of syndromes and the rapid increase in syndrome space upon using codes in large spaces. Both of these are somewhat useful in their current forms, however, with some additional mathematical work could be turned into potentially very powerful tools for the protection of quantum information. We finish off by, and along the way, mentioning various future directions to carry work on this.

Acknowledgements

I would like to thank the following people for help along the way in various forms, including encouragement, listening to ideas, and pointing out some issues to consider: Shayan Majidy, Soumik Ghosh, Rich Rademacher, Brad van Kasteren, Vinodh Raj Rajagopal Muthu, Ian George, and many others. Also, a hearty thanks goes to Andrew Jena for bouncing ideas with me and debating some of the wacky ideas proposed along the way.

I would also like to thank David Cory for him allowing me the freedom to pursue these topics–while also juggling research more strongly tied to his normal expertise. Much of this work began as weekend projects, but eventually I was granted some permission to work on these ideas during regular time too, and that allowed them to flourish. I have also learned to greatly appreciate his more purpose driven mindset: theory is nothing without experiment (or at least an experimental direction to apply it).

I would also like to thank Seth Lloyd for initially teaching me quantum computing and Isaac Chuang for introducing me to many more aspects of quantum computing–including quantum error-correction. I would also like to thank Aram Harrow for recommending Waterloo and providing guidance at times throughout my undergraduate time. I would also like to very strongly thank Daniel Gottesman and Beni Yoshida for their excellent course on quantum error-correcting codes and fault-tolerance, which equipped me with many of the basic tools needed to perform this research and the ability to understand papers in this area.

Lastly, I gratefully thank the financial contributions of the Canada First Research Excellence Fund, Industry Canada, CERC (215284), NSERC (RGPIN-418579), CIFAR, and the Province of Ontario.

Dedication

This is dedicated to all of the fantastic teachers I've had over the years. Teachers who have believed much more in me than I tended to in myself. Of particular note is David Derbes. My favorite teacher and a true role model. He is a large part of why I strive for such an interdisciplinary mastery, fostering growth among others, and remaining humble and curious.

Of course, beyond that, I must thank some of my family. Namely, my grandmother, younger sister, and older brother. They allowed me to focus on learning often times, communicating science more effectively to more people, and also to tackle life's many challenges. Thank you.

Table of Contents

Chapter 1

Background and Definitions for Qudit Codes and Error-Correction

1.1 Background

Having computational information conveyed without errors is an incredibly important problem. The ability to perform classical computation within an arbitrarily small error rate was shown by Shannon in the 40's [16]. He provided a theoretical framework showing that modern classical computation would be possible. From that point, there arose a new challenge of finding actual codes that could implement Shannon's result. This in turn pushed coding theory into a new realm, inspiring codes such as the Hamming code family and BCH codes [4], and later leading to incredible ideas such as Polar codes [2] and Turbo codes [3].

As computational power progressed, there began to be investigations into the potential power of using quantum phenomenon as a computational tool. This brought those same questions explored for classical computers back into question. These have been explored extensively, but still have plenty of directions to go. One of the major distinctions of quantum error-correction from classical error correction includes the issue of there being two error axes: bit-flip and phase-flip. No longer did it suffice to protect against bit-flips like classical codes. Although Hadamard gates could flip whether a particular error was a bit-flip or a phase-flip[1], still both axes need to be protected. This led to various ideas to try

[1] If H is the Hadamard gate given by the operator such that $HXH = Z$ and $HZH = X$, this implies that the error axes can be flipped

to bring over classical codes. Among some of the earlier ideas was the stabilizer formalism [9], CSS codes [5][19], and teleportation [11]. Many classical coding theory theorems have been generalized into this new quantum setting, such as MacWilliams' Identity for dual codes [17], Polynomial codes (a generalization of BCH and cyclic codes) [1][7], Polar codes [21], Turbo codes [15][20], Hamming bound, Gilbert-Varshamov bound, Singleton bound, as well as many other classical coding theory ideas–including results such as a complete list of all perfect codes [14].

There is at least one family of codes that exist only for qudit systems, and so this provides hints as to there being other codes for qudits only. Maximally Distance Separated (MDS) codes are a family of codes generated with a tunable parameter trading off the number of qudits being transmitted for distance [12]. This is an example of a family that only exists for qudits with at least 3 levels. This brings up the question of when else qudits might outperform qubits or at least provide more freedom than qubits.

Although many ideas from classical coding theory carries over (with minor modifications), some of these have yet to be carried over or only arise in the quantum version of coding theory. In this thesis we explore two aspects of this in particular. Firstly, can we apply codes in smaller dimensional spaces to higher-dimensional spaces? And secondly, can we utilize this result along with insights into bounds on qudit codes to generate new tools for transmission of information? We show these.

First is the ability to apply quantum error-correcting codes in smaller dimensional spaces on systems with larger alphabets without having to discover codes for those systems through other methods, thus creating extensions of these already known codes into larger spaces. Secondly, we explore how to encode classical information along with our quantum information while still maintaining protection from errors and utilizing the syndrome space of our code–these are known as hybrid codes.

Before we move on to discussing these problems, we must first define our mathematical language for working on these problems.

1.2 Definitions

In this section we define the majority of the tools used in this thesis. We recall common definitions and results for qudit operators.

A qubit is defined as a two level system with states $|0\rangle$ and $|1\rangle$. We define a qudit as being a quantum system over q levels, where q is prime. Unless otherwise specified, we require $q \geq 3$.

Definition 1. *Generalized Paulis for a space over q orthogonal levels, where we assume q is prime, is given by:*

$$\omega = e^{2\pi i/q}, \quad X_q|j\rangle = |(j+1) \mod q\rangle, \quad Z_q|j\rangle = \omega^j|j\rangle \tag{1.1}$$

where $j \in \mathbb{Z}_q$. These Paulis form a group, denoted $\mathbb{P}_q$[6].

When $q = 2$, these are the standard qubit operators. This group structure is preserved over tensor products since each of these Paulis has order q.

Definition 2. *An n-qudit stabilizer s is an n-fold tensor of generalized Pauli operators, such that there exists at least one state, $|\psi\rangle$ such that:*

$$s|\psi\rangle = |\psi\rangle \tag{1.2}$$

where $|\psi\rangle \in \mathbb{C}^{qn}$.

Definition 3. *A stabilizer group* **S** *with generators $\{s_i\}$ is defined as the subgroup of all n-qudit generalized Paulis formed from all multiplicative compositions ($\circ$) of these generators.*

Definition 4. *We call the set of states $|\psi\rangle$ which are unchanged by the stabilizer group* **S** *the codewords of the stabilizer.*

Measuring the eigenvalues of the members in our stabilizer group, called the *syndrome*, of our state gives us a way to determine what error might have occurred and then undo the determined error. This is just like the classical method of bit check syndromes.

A collection of these n-qudit stabilizers s that commutes with each other element, and still leave at least a single state, form a subgroup of all the generalized Paulis over n qudits, where each element will have order q. This forms what we call the stabilizer group **S** for these stabilizers. A collection of k compositionally independent generators for this stabilizer group will have q^k elements. We recall for the reader, the well-known result:

Theorem 5. *For any stabilizer code with k qudit stabilizers and n physical qudits, there will be q^{n-k} stabilizer states.*

For this work, we assume that all the states in our qudit system are degenerate and so all powers of Pauli operators are equally likely to occur as errors, a so-called *egalitarian* error model. With this, we have the following definition:

Definition 6. *The weight of an n-qudit operator is given by the number of non-identity operators in it.*

Definition 7. *A stabilizer code, specified by its stabilizers and stabilizer states, is characterized by a set of values:*

- *n: the number of qudits that the states are over*
- *$n-k$: the number of encoded (logical) qudits, where k is the number of stabilizers*
- *d: the distance of the code, given by the lowest weight of an undetectable generalized Pauli error (commutes with all stabilizer generators)*

These values are specified for a particular code as: $[[n, n-k, d]]_q$, where q is the dimension of the qudit space.

Now we present a couple of examples to solidify our above definitions, before moving on to a few more needed definitions. We note that, so long as no ambiguity exists, we suppress $\otimes$. We only include $\otimes$ to make register changes explicit.

Example 8. *A simple example of a code is the code generated by XX, denoted $\langle XX \rangle$, on qubits. This code has a single generator, so it trivially commutes with the entire stabilizer group $\{XX, II\}$. The code is over $n = 2$ qubits, so it has $n - k = 1$ encoded qubits, so we have only a pair of stabilized codewords, namely:*

$$|0\rangle_l = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \quad |1\rangle_l = \frac{|01\rangle + |10\rangle}{\sqrt{2}} \tag{1.3}$$

where $|0\rangle_l$ and $|1\rangle_l$ are the encoded logical codewords. The distance of this code is $d = 1$ since any single qubit X operator commutes with the whole space but is not in the stabilizer group. Lastly, to make this operational, we would want a logical X and a logical Z operator. X on the first register carries us between the states and Z on the first register applies a phase to the $|1\rangle_l$ state, and these operators satisfy standard X, Z commutation relations. Then we say that the code generated by the stabilizers $\langle XX \rangle$ has parameters $[[2, 1, 1]]_2$. This is a poor example of a code, but is still technically valid.

Example 9. *Our next example is the code generated by $\langle XX, ZZ \rangle$ again on qubits. The total stabilizer group is given by $\{XX, ZZ, XZ \otimes XZ, II\}$. These all commute since $[XX, ZZ] = 0 \mod 2$. This code is again over $n = 2$ qubits, but now has $n - k = 0$ encoded qubits, so this means that we have a single stabilized state, namely:*

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{1.4}$$

This is a valid code, and this codeword is the intersection of the code spaces of the codes generated by $\langle XX \rangle$ and $\langle ZZ \rangle$. Since there's a sole codeword, there is no valid logical operator for X and Z. However, we note that for this codeword all single qubit operators don't commute with the generators, so the distance of this code is now $d = 2$. We notice that we've made a trade-off of sorts–sacrificing an encoded qubit for an increase in the distance of the code. We can write the parameters of this code as $[[2, 0, 2]]_2$.

With these two examples, we begin to see that in stabilizer codes certain trade-offs occur in many cases. We now provide our last few definitions that will be used throughout the thesis.

Definition 10 (Egalitarian Error Model)**.** *The error occurring on a system is defined as egalitarian if all errors of a constant weight are equally likely to occur.*

This means that Z and X^{-1} and XZ^{-1} all occur with the same probability, as do all other single qudit errors. We assume that from this, all weight two errors occur with the same probability. This is a reasonable error model since we could simply take the dominant error and overestimate the rate of the other types of errors using that.

Working with tensors of operators can be challenging, and so we make use of the following well known mapping from these to vectors. By representing these operators as vectors at times the solution to a problem can become far more tractable.

Definition 11 (ϕ representation of a qudit operator)**.** *We define the surjective map*

$$\phi_q : \mathbb{P}_q^n \mapsto \mathbb{Z}_q^{2n} \tag{1.5}$$

which carries an n-qudit Pauli in $\mathbb{P}_q^n$ to a $2n$ vector mod q, where we define this map as:

$$\phi_q(\omega^\alpha \otimes_{i-1} I \otimes X_p^a Z_p^b \otimes_{n-i} I) = (0^{\otimes(i-1)}\ a\ 0^{\otimes(n-i)} | 0^{\otimes(i-1)}\ b\ 0^{\otimes(n-i)}) \tag{1.6}$$

This mapping is also a homomorphism if we define: $\phi_q(s_1 \circ s_2) = \phi_q(s_1) \oplus \phi_q(s_2)$, where $\oplus$ is addition mod q. We denote the first half of the vector as $\phi_{q,x}$ and the second half as $\phi_{q,z}$.

We may invert the map ϕ_q to return to the original n-qudit Pauli operator with the leading coefficient being undetermined, however, this generally preserves code properties. We make note of a special case of the ϕ representation.

Definition 12. *Let q be the dimension of the initial system. Then we denote by ϕ_∞ the mapping:*

$$\phi_\infty : \mathbb{P}_q^n \mapsto \mathbb{Z}^{2n} \tag{1.7}$$

where no longer are any operations taken mod, but instead carried over the integers.

This definition follows the previous one, since any number $\mod \infty$ is just the number itself. In general we will write a stabilizer as ϕ_q, perform some operations, then write it in ϕ_∞. We shorten this to write it as ϕ_∞, and can later select to write it as $\phi_{q'}$ for some prime q' by taking element-wise $\mod q'$. When we provide no subscript for the representation, that implies that the choice is irrelevant.

Example 13. *We revisit our prior example of the code generated by $\langle XX, ZZ\rangle$, denoted Ξ here. We assume this is a qubit code and so:*

$$\phi_2(\Xi) = \begin{bmatrix} 1 & 1|0 & 0 \\ 0 & 0|1 & 1 \end{bmatrix}$$

This is the ϕ_2 representation for the code generated by Ξ. We can append to the end $(1, 1|1, 1)$ to also include $XZ \otimes XZ$, however this is linearly dependent on the other rows, so we exclude it here.

The commutator of two operators in this picture is given by:

Definition 14. *Let s_i, s_j be two qudit Pauli operators over q bases, then these commute if and only if:*

$$\phi_q(s_i) \odot \phi_q(s_j) = 0 \mod q \tag{1.8}$$

where $\odot$ is the symplectic product, defined by:

$$\phi_q(s_i) \odot \phi_q(s_j) = \oplus_k [\phi_{q,z}(s_j)_k \cdot \phi_{q,x}(s_i)_k - \phi_{q,x}(s_j)_k \cdot \phi_{q,z}(s_i)_k] \tag{1.9}$$

where $\cdot$ is standard integer multiplication mod q and $\oplus$ is addition mod q.

Before finishing, we make a brief list of some possible operations we can perform on our ϕ representation for a stabilizer group:

1. As remarked above, we may add rows together, which corresponds to composition of operators
   (a) If this operation preserves the rank of ϕ, then this is also equivalent to selecting alternative generators
2. We may swap rows, corresponding to permuting the stabilizers
3. We may multiply each row by any number in $1 \to q-1$, corresponding to composing a stabilizer with itself. Since all operations are done over a prime number of bases, each number has an inverse.

4. We may swap registers (qudits) in the following ways:

   (a) We may swap columns $(Reg\ i, Reg\ i+n)$ and $(Reg\ j, Reg\ j+n)$ for $0 < i, j \leq n$, corresponding to relabelling qudits.

   (b) We may swap columns $Reg\ i$ and $(-1) \cdot Reg\ i+n$, for $0 < i \leq n$, corresponding to conjugating by a Hadamard gate on register i (or Discrete Fourier Transforms in the qudit case [10]) thus swapping X and Z's roles.

All of these operations leave all properties of the code alone, but can be used in proofs.

At this point we have all the necessary definitions to prove our results and have a solid base in qudit operators.

Chapter 2

Embedding Stabilizer Codes and the Stabilizer Hierarchy

The promise of quantum computing seems great, but overcoming the challenges of errors on such systems has been an ongoing problem since its conception. One well studied proposal involves the quantum analog of classical error correcting codes, known as stabilizer codes. Classical error correcting codes can be applied across higher base computing, such as trinary, but it has been unknown whether this result carries over for stabilizer codes. In this chapter we show that this result carries at least for sufficiently large base dimension of the quantum system. The implications of this is that stabilizer codes can immediately be carried over to higher dimensions, but that there exist codes in these higher qudit (a qubit with possibly more than 2 bases) spaces which might allow for better parameters–or other error correction advantages of qudits over qubits, which remained an unsolved problem.

Aside from this, another crucial utility is being able to rapidly produce qudit codes without having to search for them. We can instead apply known stabilizer codes onto larger spaces, although perhaps sacrificing some of the potential transmission rate. As qudit quantum computing devices begin to become a reality, the ability to carry over codes directly may prove a valuable resource, and might aid in determining if a code is packing information better by utilizing the higher dimensionality.

The format of this chapter is as follows. We begin by defining *invariant* codes, which are the codes which can be embedded, then proceed to show that all qudit codes are invariant codes over larger spaces. This only shows that codes are valid over higher spaces, we then show that at least for sufficiently sized spaces, all parameters of the code–particularly the distance–is at least preserved, if not even improved. We provide an argument about

when the distance of the code will be improved upon embedding. Along with these we provide a bound on the number of bases considered sufficient for this problem–although propose as an important continuation showing that the distance is at least preserved for all larger prime dimension spaces. With these codes shown, we proceed to provide an explicit way of constructing the embedded codewords, reducing the complexity of determining the codewords for these codes and providing an alternate way to interpret these codes. Having all this shown, we reach our conclusion and suggest future directions.

2.1 Embedding Theorem

In this section we define and prove an embedding theorem in two steps: first showing that all codes can be embedded into larger spaces, and second by showing that all parameters of the code are at least preserved in sufficiently high dimensions. The embedding theorem then allows us to transform qudit codes into forms that can be used as qudit codes in larger spaces while still keeping all the parameters of the original code.

We begin by defining what property all embedded codes need to satisfy:

Definition 15 (Invariant codes)**.** *A stabilizer code is invariant iff:*

$$\phi_q(s_i) \begin{bmatrix} & -I_n \\ I_n & \end{bmatrix} \phi_q(s_j)^T \equiv 0 \mod q, \quad \forall i, j$$

holds for all primes q.

This can equivalently be stated as $\phi_\infty(s_i) \odot \phi_\infty(s_j) = 0$, for all stabilizers s_i and s_j in the stabilizer group $\mathbf{S}$.

2.1.1 Motivating Examples

Consider the following example of generators for a stabilizer group: $\langle XX, ZZ \rangle$. As a qubit code this forms a valid stabilizer code with codeword:

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{2.1}$$

and the commutator of these generators can be seen to be: $(1) + (1) = 2 \equiv 0 \mod 2$. Now suppose we wish to use this code for a qutrit system. In order to do that we must

transform these generators into ones which have commutator 0, this can be achieved with $\langle XX^{-1}, ZZ\rangle$. In this case $\phi(X \otimes X^{-1}) \odot \phi(Z \otimes Z) = 0$. This means that not only can this be used for qutrits, but for all prime number of bases. The codeword in the qutrit case is:

$$\frac{|00\rangle + |12\rangle + |21\rangle}{\sqrt{3}} \tag{2.2}$$

and the generalization of this for the codewords is a simple extension. We would simply make each codeletter in the codeword have the entries sum to a multiple of the qudit dimension so that the ZZ operator has a $+1$ eigenvalue:

$$\frac{1}{\sqrt{q}}(\sum_{j=1}^{q} |j \mod q, q-j \mod q\rangle) \tag{2.3}$$

If we look at the generators of this code, there is no single qudit operator that commutes with the generators, thus the distance of this invariant form of the code is still $d = 2$.

This is not the only example of a code that can be turned into invariant form. Another great example is the 5-qubit code. In fact, we don't even need to make any changes:

$$\langle XZZXI, \quad IXZZX, \quad XIXZZ, \quad ZXIXZ\rangle \tag{2.4}$$

From inspection this can be seen to have commutators 0, and so this is a valid stabilizer code for qudits, and it can also be checked that this code will always have distance 3.

It is helpful to have a couple of examples, however, it has been unknown whether it is always possible to put stabilizer codes into invariant form. We move forward from here to show that this can be done, and how to do this. We also show that for infinitely many primes the distance of the code is at least preserved when applied to larger spaces.

2.1.2 Embedding Theorem Statement and Proof

We now show that all qudit stabilizer codes can be written in an invariant form[1]. This shows that we can apply these codes directly over any number of bases, but says nothing about the distance of these codes. This aspect is treated in the theorem immediately following.

Theorem 16. *All qudit stabilizer codes can be transformed into invariant codes.*

[1]We acknowledge Andrew Jena for his contributions in the form of the below theorem and corollary.

Proof. Let $\{s_1, \ldots, s_k\}$ be a basis of qudit stabilizers for a code, S, with $k \leq n$ and a prime, q. We must construct a set of stabilizers, $\{s'_1, \ldots, s'_k\}$, such that:

1. $\phi_\infty(s'_i) \equiv \phi_q(s_i) \mod q$, for all i
2. $\phi_\infty(s'_i) \odot \phi_\infty(s'_j) = 0$, for all $i \neq j$.

Without loss of generality, we assume that our stabilizers are given in canonical form:

$$\begin{pmatrix} \phi_q(s_1) \\ \vdots \\ \phi_q(s_k) \end{pmatrix} = \left(\begin{array}{cc|cc} I_k & X_2 & Z_1 & Z_2 \end{array} \right).$$

We define the strictly lower diagonal matrix, L, with entries:

$$L_{ij} = \begin{cases} 0 & i \leq j \\ \phi_\infty(s_i) \odot \phi_\infty(s_j) & i > j \end{cases}$$

and define $s'_1, \ldots, s'_k$ such that:

$$\begin{pmatrix} \phi_\infty(s'_1) \\ \vdots \\ \phi_\infty(s'_k) \end{pmatrix} = \left(\begin{array}{cc|cc} I_k & X_2 & Z_1 + L & Z_2 \end{array} \right).$$

We show that $s'_1, \ldots, s'_k$ satisfy the conditions.

1. Since $\phi_q(s_i) \odot \phi_q(s_j) \equiv 0$ for all $i \neq j$, we observe that $L \equiv 0_k \mod q$. By adding rows of L to our stabilizers, we have not changed the code modulo q.
2. For $i > j$, we observe that:

$$\begin{aligned} &\phi_\infty(s'_i) \odot \phi_\infty(s'_j) \\ &= (\phi_\infty(s_i) + (0 \mid L_i \; 0)) \odot (\phi_\infty(s_j) + (0 \mid L_j \; 0)) \\ &= \phi_\infty(s_i) \odot \phi_\infty(s_j) + \phi_\infty(s_i) \odot (0 \mid L_j \; 0) \\ &\quad + (0 \mid L_i \; 0) \odot \phi_\infty(s_j) + (0 \mid L_i \; 0) \odot (0 \mid L_j \; 0) \\ &= \phi_\infty(s_i) \odot \phi_\infty(s_j) + 0 - \phi_\infty(s_i) \odot \phi_\infty(s_j) + 0 \\ &= 0. \end{aligned}$$

□

Example 17. *Consider the 7-qubit Steane code with parameters* $[[7,1,3]]_2$*, denote it by* Ξ *[18]. The* ϕ *representation is given by:*

$$\phi_2(\Xi) = \left[\begin{array}{c|c} H & 0 \\ 0 & H \end{array}\right]$$

where H *is the parity-check matrix for the classical Hamming code given by:*

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

We begin by putting this in standard form. First we apply Hadamards on registers 4,5, and 7, then diagonalizing those rows, this is:

$$\phi_2(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{array}\right]$$

Next, performing addition $\mod 2$*, we perform the following row operations:* $r4 \leftarrow r4+r5$*,* $r6 \leftarrow r6+r4$*,* $r5 \leftarrow r5+r4+r6$*. This then provides our matrix as:*

$$\phi_2(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

Now we swap registers: $(reg5, reg4)$*,* $(reg6, reg7)$*. Now our matrix is:*

$$\phi_2(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

We remind the reader that none of the these operations need to be actually applied to the system. These operations are simply mathematically applied, then, after being put into invariant form, we may undo these operations to obtain how the original generators have been altered. In short, this method requires no operations performed on the system, just altering the choice of the generators. Our code Ξ is now in standard form. For the following operations, we no longer take our operations over $\mod 2$.

The following is the anti-symmetric matrix $[\odot]$ representing the commutators between the stabilizers and the L_{ij} matrix for this code:

$$[\odot] = \begin{bmatrix} 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \Rightarrow L_{ij} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Adding this to our standard form, we have an invariant form for the Steane code given by:

$$\phi(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

We will want to know the size of the maximal entry in this invariant form for our bound on ensuring the distance of the code is at least preserved. The bound on the maximal entry is provided from the above proof:

Corollary 18. *The maximal element in $\phi_\infty(S)$ is upper bounded by:*

$$B = (2 + (n-k)(q-1))(q-1) \tag{2.5}$$

Proof. For any $i \neq j$, there are at most $n-k$ entries in which both $\phi_q(s_i)$ and $\phi_q(s_j)$ are non-zero and bounded above by $q-1$, and a single entry in which one is 1 whereas the other is bounded above by $q-1$. This gives us a bound on the inner product of: $(n-k)(q-1)^2 + (q-1)$. This is a bound on the size of an entry in our invariant stabilizer of $q - 1 + (n-k)(q-1)^2 + (q-1) = (2 + (n-k)(q-1))(q-1)$. □

Example 19. *In this example we show that CSS codes remain CSS codes under this transformation. Consider a generic CSS code given by:*

$$\phi(\Xi) = \left[\begin{array}{ccc|ccc} I_{k_1} & X_{k_2} & X_{n-(k_1+k_2)} & 0 & 0 & 0 \\ 0 & 0 & 0 & Z_{k_1} & I_{k_2} & Z_{n-(k_1+k_2)} \end{array}\right]$$

where we have put the two block matrices into approximately standard form. Now, we perform Hadamards (or discrete Fourier transforms) on the k_2 sized middle blocks. We then have:

$$\phi(\Xi) = \left[\begin{array}{ccc|ccc} I_{k_1} & 0 & X_{n-(k_1+k_2)} & 0 & X_{k_2} & 0 \\ 0 & I_{k_2} & 0 & Z_{k_1} & 0 & Z_{n-(k_1+k_2)} \end{array}\right]$$

Now, we note that the first k_1 stabilizers commute with each other and likewise for the k_2 other stabilizer generators. Now we simply need to consider the case where we pick generators from each of the halves. We consider the matrix $[\odot]$, as above. This has nonzero entries for rows in k_2 when the columns are in k_1. Likewise for when the rows are in k_1, the entries are nonzero for columns in k_2, thus we only add entries to Z_{k_1} and X_{k_2} with $[\odot]$ and so certainly do for our L matrix. We may now invert our initialization and we will still have a CSS code.

Now that we know that all qudit codes can be put into an invariant form, we now prove that at least for most sizes of the space we can ensure that the distance of the code is at least preserved. We can explicitly construct sizes of the underlying space that will at least preserve the distance.

Theorem 20. *There exist infinitely many primes $p > p^*$, with p^* a cutoff value, such that the distance of a non-degenerate stabilizer code $[[n, n-k, d]]_p$ still has at least the same distance.*

Before proving this theorem, we make a couple of definitions:

Definition 21. *An unavoidable error is an error that commutes with all stabilizers and produces the $\vec{0}$ syndrome over the integers*

Remark 22. *The distance of a code over the integers is given by the minimal weight member in the set of unavoidable errors. The distance over the integers is represented by d^*, and so $d^* \geq d$.*

This value is also the number of columns that are linearly independent over the integers (or equivalently, over the rationals).

Definition 23. *An artifact error is an error that commutes with all stabilizers but produces at least one syndrome that is only zero modulo the base.*

Proof. We recall that the ordering of the stabilizers and the ordering of the registers does not alter the distance of the code. With this, ϕ_∞ for the stabilizer generators over the integers can have the rows and columns arbitrarily swapped.

Let us begin with a code over q bases and extend it to p bases. The errors for the original code are the vectors in the nullspace of $S_q \mod q$. These errors are either unavoidable errors or are artifact errors. We may rearrange the rows and columns so that the stabilizers and registers that generate these nonzero entries are the upper left $2d \times 2d$ minor, padding with identities if needed. The factor of 2 occurs due to weights in ϕ_∞ are up to double those of the Pauli. The stabilizer(s) that generate these multiples of q entries in the syndrome are members of the null space of the minor formed using the corresponding stabilizer(s).

Now, consider the extension of the code to p bases. Building up the qudit Pauli operators by weight j, we consider the minors of the matrix composed through all row and column swaps. These minors of size $2j \times 2j$ can have a nontrivial null space in two possible ways:

- If the determinant is 0 over the integers then this is either an unavoidable error or an error whose existence did not occur due to the choice of the number of bases.
- If the determinant is not 0 over the integers, but takes the value of some multiple of p, then it's $0 \mod p$ and so a nullspace exists.

So we can only introduce artifact errors to decrease the distance. By bounding the determinant by p^*, any choice of $p > p^*$ will ensure that the determinant is a unit in $\mathbb{Z}_p$, and hence have a trivial nullspace since the matrix is invertible.

Now, in order to guarantee that the value of p is at least as large as the determinant, we can use Hadamard's inequality to obtain:

$$p > p^* = B^{2(d-1)}(2(d-1))^{(d-1)} \tag{2.6}$$

where B is the maximal entry in ϕ_∞. Since we only need to ensure that the artifact induced nullspace is trivial for Paulis with weight less than d, we used this identity with $2(d-1) \times 2(d-1)$ matrices.

When $j = d$, we can either encounter an unavoidable error, in which case the distance of the code is d or we could obtain an artifact error, also causing the distance to be d. It is possible that neither of these occur at $j = d$, in which case the distance becomes some d' with $d < d' \leq d^*$. □

Example 24. *In our example of the Steane code, we then have $B = 1$ and $d = 3$, so for all primes larger than $1^{2\cdot 2}(2 \cdot 2)^2 = 16$ we are guaranteed that the distance is preserved. For primes below that value, we can manually check and apply alternate manipulations if needed. Given the structure of the matrix, we know that the determinant of all the minors is bounded by 4, all primes at least as large as 5 preserve the distance and through manual checking 3 also works, so all primes preserve the distance for our invariant form of the Steane code.*

Remark 25. *We note that our result carries over to more than stabilizer codes. These results should carry over to all quantum codes that can be represented by a matrix, with minor alterations to p^*.*

We alluded prior to this proof that the code over the integers has distance at least as large. This begs the question of at what point will this distance d^* be ensured? To this, we simply extend our above result slightly to obtain the cutoff expression, whereby no further distance improvements can be obtained from embedding the code–suggesting that another code ought to be used.

Corollary 26. *We obtain the integer distance d^* when:*

$$p > B^{2(d^*-1)}(2(d^*-1))^{d^*-1} \tag{2.7}$$

after this value, the distance cannot be improved through embedding. If d^* is unknown, this can be upper bounded by using k in place of d^*.

Proof. This follows from the above proof. The looser bound comes from $d^* \leq k$, so we can evaluate this at $d^* = k$ to obtain the loosest condition. □

2.1.3 The Hierarchy of Stabilizer Codes

Definition 27. *Denote the set of all stabilizers codes over n qudits over q levels as $\mathcal{S}_n[\mathbb{C}^q]$ and the set of all qudit stabilizer codes as: $\mathcal{S}[\mathbb{C}^q] = \cup_{n=1}^{\infty}\mathcal{S}_n[\mathbb{C}^q]$.*

The previous proofs tell us that all qudit codes can be embedded into qudit codes over larger spaces, however, it says nothing about inscribing them into smaller spaces. We address this now, showing that it cannot always be done while preserving the distance of the code.

Theorem 28 (Inscribing Theorem)**.** *Let all stabilizers be written in invariant form, and let $p < q$ be primes, then:*

$$\mathcal{S}_n[\mathbb{C}^p] \prec \mathcal{S}_n[\mathbb{C}^q] \tag{2.8}$$

and moreover:

$$\mathcal{S}[\mathbb{C}^p] \prec \mathcal{S}[\mathbb{C}^q] \tag{2.9}$$

where $\prec$ indicates that there are some codes for which $[[n, n-k, d]]_q$ become $[[n, n-k, d']]_p$ with $d' < d$.

This shows that the quality of a code won't go down upon being embedded (at least for sufficiently large spaces), but might if inscribed. Of particular note is the case where $p = 2$ where this implies that we can apply qubit codes in larger spaces and that there are some qudit codes which will decrease in distance upon being inscribed.

Proof. Consider the following fragments of stabilizer generators over n registers for any integer choice of p:

$$\alpha = \langle X^p X, Z Z^p \rangle, \quad \beta = \langle Z^p X^p Z, X Z^2 X^p \rangle \tag{2.10}$$

these fragments α and β are both invariant and so concatenation (tensor product of registers) of these will still form an invariant code. Thus strings such as:

$$\alpha\beta\alpha\alpha\alpha\alpha, \quad \alpha\beta\alpha\beta\beta \tag{2.11}$$

are valid pairs of stabilizer generators over $n = 13$ registers. Since any integer n is either even or odd, we can write a valid code of length n as either: 1 β and $\frac{n-3}{2}$ α or as $\frac{n}{2}$ iterations of α. This is not all codes of length n, but is a valid code and in invariant form. In addition, α has distance $d = 2$ so long as we don't have p bases, otherwise it has distance 1 only. Therefore, the distance of this code decreases whenever we have p bases. Now, we may select all primes below q as stabilizers of this form and so we know that there will always exist stabilizers whose distance can decrease upon being inscribed into a smaller space. □

This then suggests a hierarchy for the stabilizers of qudit operators: all qubit codes are able to be transformed into invariant forms for dimension two or greater, and all qutrit codes can be used for qudits with dimension three or greater, and all qudit codes with five levels can be used for qudits with dimension five or greater, and so forth. In order to show this, we must ensure that embedding can never decrease the distance of a code, an extension of the work here. If such a hierarchy is shown it might suggest that as the underlying space increases in size it opens more codes up as options, perhaps allowing for additional nice properties and rates not priorly possible for lower dimensional codes.

Remark 29. *Unfortunately, we have only been able to show a weaker version of this where $q > p^*$, but believe this to always be true. As an aid, we provide the runtime for determining the distance of an invariant form of a code.*

```
for t in range(1,d*-1):
    select t registers of the n, call this partition y
        for each element in y vary through 1 to q^2-1
            if y is in the nullspace of S, then d=t
                return d
return d*
```

Runtime: $O((d^*-1)\binom{n}{(d^*-1)}(q^2-1)^{d^*-1}) \approx O((d^*-1)(\frac{n\cdot e}{d^*-1})^{d^*-1}(q^2-1)^{d^*-1})$. *This can be written as:* $O((d^*-1)(\frac{n\cdot e\cdot q^2}{d^*-1})^{d^*-1})$. *To the most important order, we can estimate this as* $O((nq^2)^{d^*-1})$. *This is somewhat slow, but not unreasonable for distance determination.*

Now that we know that we may embed codes, we now show that we can immediately and explicitly construct the codewords for these embeddings.

2.2 Explicit Construction of the Embedded Codewords

In this section we describe how to generate the codewords for a code that has been embedded using the stabilizer generators and a single element within each codeword. This latter restriction can be removed if all encoded X operators are known. This can be viewed as a dual interpretation of the stabilizer generators: as lattice vectors on the indices of the logical basis and operators to shift this coarse lattice to fill the entire space.

The traditional way to determine the codewords for q bases is to project onto the stabilizer group using:

$$\Pi = \prod_i \left(\sum_{j=0}^{q-1} s_i^j \right) \tag{2.12}$$

This method: 1) requires multiplication and 2) needs to know q to determine this projector. It could be powerful to have an additive method that doesn't require knowledge of q. We now show this alternative way of looking at the codewords.

Definition 30. *The simultaneous weight of an n-qudit operator over q levels, s, is given by:*

$$\xi := \xi(s) = \phi_{q,x}(s) \cdot \phi_{q,z}(s) \tag{2.13}$$

Definition 31. *For some stabilizer code* $\mathbf{S}$*, let* C *be an orthonormal* $+1$ *eigenstate of this stabilizer group. Each* C *is a codeword. Each* C *is a superposition of states in the computational basis, termed codeletters* c*:*

$$C = \sum a_c |c\rangle \tag{2.14}$$

with $a_c \in \mathbb{C}$ *and* $\|a\|^2 = 1$*, where* $a_c = 0$ *is allowed.*

The codeletters c *may be represented by an* n*-digit* q*-ary number, or equivalently as a vector in* $\mathbb{Z}_q^n$*.*

A Pauli operator only has X, Z operators, so the ϕ representation of these operators will be a vector in $\mathbb{Z}_2^{2n}$, where each entry in the vector corresponds to the power of the operator in the Pauli as given by ϕ_q. Likewise, for an n-qudit operator, the ϕ_q representation of an operator will be a vector in $\mathbb{Z}_q^{2n}$.

The action of a stabilizer on each codeletter c is to either map it to itself again or to map it to another codeletter in the same codeword with perhaps a different coefficient.

We begin by proving a lemma about the structure of the codewords in terms of their codeletters.

Lemma 32. *Let* $\mathbf{S}$ *be a stabilizer group and* $\{C_m\}$ *be the codewords for this code and the logical* Z *operators have disjoint support. Let* k *be the number of generators for* $\mathbf{S}$ *over* q *levels and denote each generator by* s_i*. If the stabilizer group* $\mathbf{S}$ *has logical* Z *operators that have disjoint supports and are solely composed of* Z *operators, then:*

- *Each codeletter has length* n*, and these can be written so that there are* q^k *code letters in each codeword–allowing for coefficients of 0 where needed.*
- *Each codeword is formed by* q^{k-1} *disjoint cycles of length* q*, or by* q^k *disjoint cycles of length 1 under the action of* s_i*.*

Proof. From theorem 5, there are q^{n-k} code words and the entire space has size q^n, so each codeword has q^k codeletters. Each codeletter has length n since the code is over n qudits.

Next, any generator for $\mathbf{S}$ has order q. Let s be some generator for $\mathbf{S}$. If $\phi_{q,x}(s) \neq 0$, then $sc \propto c'$ and $s^q c = c$, for a pair of codeletters c, c'. This implies that C can be broken into cycles of length q. Next, we note that in this case, the cycles must be formed from unique codeletters, since we have logical Z operators that are products of Z operators only, and thus are disjoint. This implies that there are q^{k-1} disjoint cycles of length q.

Next, we consider the case that $\phi_{q,x}(s) = 0$. These are the generators that are composed solely of Z operators. These cause the codeletters to form cycles of length 1, $sc = c$ or $a_c = 0$, and thus all the codeletters are disjoint.

Lastly, these cycles mix upon taking elements from $\mathbf{S}$ that are compositions of generators, however, the resulting behavior still follows. □

Definition 33 (XZ Form)**.** *All n-qudit stabilizers, s, over q levels can be written in the form:*

$$s = \omega_q^r \otimes_{j=1}^n X_q^{a_j} Z_q^{b_j} \tag{2.15}$$

for some $a, b, r \in \mathbb{Z}_q$. This is the XZ form for the stabilizer s.

Theorem 34. *Let $\mathbf{S}$ be a stabilizer group with generators $\{s_i\}_i$ for a stabilizer code, where it has been written over the integers, and let $\{X_t\}_t$ be the set of logical X operators for this code. Then we can write the codeletters in each codeword as vector additions to the all 0 codeletter and applications of X_t.*

This says that we do not need to preemptively know the dimension of the underlying space but can still determine the codewords, and in an efficient manner.

Proof. Assume the stabilizer s is in XZ form, then the action of this stabilizer on a codeletter cycle c_l is given by:

$$s \sum_l \omega^{a_l} |c_l\rangle = s \sum_l \omega^{a_l} |c_0 + l\phi_x(s)\rangle \tag{2.16}$$

Applying only the Z portion of the operator, we collect a phase from those registers that are not altered by the X portion, denoted $\phi_0(s)$, as well as $\xi(s)$ phase from the simultaneous operators:

$$\sum_l \omega^{a_l + \phi_0(s) + \xi} |c_0 + (l+1)\phi_x(s)\rangle \tag{2.17}$$

which means that in order for this cycle to be stabilized, we must have:

$$a_1 = a_0 + \phi_0(s), \quad a_2 = a_1 + \xi = a_0 + \phi_0(s) + (2-1)\xi, \quad a_l = a_0 + \phi_0(s) + (l-1)\xi \tag{2.18}$$

This gives us an equation set that each cycle must satisfy for this stabilizer.

We wish for there to be coefficients to satisfy our condition above for each of the k stabilizers. Before proceeding, we note a reduction:

$$\begin{aligned} a_q = a_0 &= a_0 + \phi_0(s) + (q-1)\xi \\ &\Rightarrow \phi_0(s) + (q-1)\xi = 0 \\ &\Rightarrow \phi_0(s) = \xi \end{aligned} \tag{2.19}$$

with this conclusion, generally: $a_l = a_0 + l\xi$. This generally means that we only have 1 degree of freedom per cycle unless we consider that we also have an additional freedom from taking s^b as our stabilizer instead of s, with $b \in [1, q-1]$. This is still an operator with order q and is equivalent to s being a stabilizer. Using this, we have:

$$a_l^j = a_0^j + bl\xi \tag{2.20}$$

meaning that we have 2 parameters, or in total $2k$ variables which need to satisfy k equations and each cycle may overlap with another cycle only once. Since ξ is constant for each stabilizer across all cycles, we will always have a solution so long as the original s had one.

□

The condition of the logical Z operators being composed solely of Z operators is not nearly as restrictive as first appearance. We can ensure this condition for all codes with disjoint supports for the logical Z operators.

Lemma 35. *Let g_i be the logical Z operators for a stabilizer code S, with each g_i having disjoint support. Let $g = g_i$ for some g_i, then we can write g as a product of all Z operators through Clifford conjugations, for each i.*

The Clifford group for qudits contains qudit Paulis and the following operators (these are from [10], with specification of the power of the S gate):

Definition 36. *The Hadamard is replaced by the discrete Fourier transform $\mathcal{F}$:*

$$\mathcal{F}X\mathcal{F}^\dagger = Z, \quad \mathcal{F}Z\mathcal{F}^\dagger = X^{-1} \tag{2.21}$$

Definition 37. *The phase gate $\mathcal{P}$ performs:*

$$\mathcal{P}Z\mathcal{P}^\dagger = Z, \quad \mathcal{P}X\mathcal{P}^\dagger = XZ \tag{2.22}$$

Definition 38. *The power gate S performs:*

$$SXS^\dagger = X^2, \quad SZS^\dagger = Z^{2^{-1}} \tag{2.23}$$

Proof. We work component-wise. WLOG we write the term as: $X^\alpha Z^\beta$. Then conjugating by $\mathcal{P}^{-\beta}$ then $\mathcal{F}$ turns this term to Z^α. Lastly, conjugate by $S^{2\alpha^{-1}}$. Applying this form each term, we obtain a logical Z operator that only contains Z terms. All of these Clifford operations preserve the distance of the code. □

Then so long as the logical Z operators for a code have disjoint supports then the above holds, and so we can create view the codewords in this other way. This then has provided us with another interpretation for the codewords of these embedded codes as vector motion along the computational basis, so as long as we have the logical X operators we may interpret them as turning the lattice formed by a codeword to dual lattices, forming the entire computational basis grid.

Remark 39. *Combining the results in this chapter implies the ability to define stabilizer codes for systems defined over the integers (with countably-infinite number of bases). This could be useful for future systems.*

Example 40. *We again work with the Steane code. In total there are q^7 possible codeletters, most of which will have coefficient 0. We recall our invariant form for the code Ξ:*

$$\phi(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

We perform discrete Fourier transforms, $\mathcal{F}$, on registers 1 and 7 to obtain:

$$\phi(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ -1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

We now have a stabilizer that is composed solely of Z operators.

Beginning with $|0\rangle^{\otimes 7}$ we obtain:

$$|0\rangle_l \propto \omega^{\Upsilon} |f-d, b, c, d, e, f, d+e\rangle \tag{2.24}$$

$$\Upsilon = a\cdot(f-d)+(d+e+f)\cdot b+(f-e)\cdot c+(b+c)\cdot d+(a+b)\cdot e+(a+b+c)\cdot f+(a+c)\cdot(d+e) \tag{2.25}$$

where $a, b, c, d, e, f \in \mathbb{Z}$. Having the logical X operator for this matrix would allow us to have the logical $|1\rangle$ state. Restricting all variables to $\mathbb{Z}_2$, we obtain the standard $|0\rangle_l$ codeword for the Steane code.

In this way, we now have a version of the Steane code with parameters $[[7, 1, 3]]_\infty$.

2.3 Conclusion and Discussion

Although in this work we find some critical value above which all primes preserve the distance of the code, we believe that this result carries to all primes at least as large as the initial dimension. Proving this, or at least tightening the bound on the critical value, seems like an important extension of this result, since the current bound can be quite large. In addition, it would make the hierarchy for stabilizer codes more explicit. In addition, there is the question of whether these results also hold for degenerate codes. Beyond this, there is the important question of whether there exists some property of qudit systems that could allow for some aspect of error correction not accountable for by qubits. Although at first glance, the answer would appear to be no, it is worth considering that truncating a Hilbert space by neglecting certain values might result in some defects (such as not truly having a prime number of bases) preventing the qudit advantage from appearing.

In this chapter we have achieved the following. We have shown that qudit codes can be embedded into larger spaces, and at least for sufficiently large number of bases, all properties are preserved or improved, thus implying a hierarchy of stabilizer codes whereby code from each space are a strict subset of codes possible for the larger sized space. This result may aid in error correction schemes for qudit based quantum computing by firstly providing immediate codes for these devices by applying already known codes, and secondly perhaps aiding in search for codes for these qudit devices that better utilize the underlying higher dimensionality of these systems. Here we have provided a method for producing the codewords of these embedded codes without having to compute them from scratch or using another method, but simply adding to the registers and applying an associated phase.

Some important directions to carry these results include the following. Firstly, some additional applications of this result showing additional utility beyond those discussed already could provide more avenues to improve error protection. A first application is presented in the next chapter. Secondly, upon testing many codes, we found that distance was at least preserved for embedded codes over all primes greater than or equal to the original number of bases, so long as care was taken in performing the embedding. We were not able to show this result here. However, this would be an important result, thus showing that embedding can never *harm* the distance of the code. Lastly, as alluded to earlier, this allows for immediate creation of embedded codes. However, it is not clear whether there is, or how greatly there is, an advantage of using stabilizer codes within the corresponding level of our proven hierarchy. Perhaps embedded codes are approximately equivalent to codes optimized for the number of bases in the space, however, we suspect that it's unlikely, but leave that as another open problem.

Chapter 3

Transforming Particular Stabilizer Codes into Hybrid Codes

Coding of quantum information has made a great deal of progress since its early inception. Still, there is a continuing search for the ability to communicate classical information with the quantum information without incurring additional costs. Codes which can accomplish this are called hybrid codes. Here, we propose (and prove) a scheme for encoding classical information using a specific family of hybrid codes as direct extensions of stabilizer codes. The tools proven here can also be applied to other stabilizer codes and our new families of hybrid codes provide many more hybrid codes than previously known.

Since their first discovery, stabilizer codes have proven a useful tool for protecting quantum bits (qubits) against some noise [9]. There has been some work on defining these codes over higher dimensional systems[10]. These higher dimensional systems are referred to as *qudits*, and throughout we will use q as the number of orthonormal basis states and assume q is prime. By and large, qudit codes have not been extensively studied, however. Likewise, only somewhat recently have hybrid codes begun to garner interest [8],[13]. These have either focused on finding a few explicit examples or on very broadly whether it would be possible to transmit classical information as well without reducing the quality of the quantum code. Here, we show that it is possible for some families of stabilizer codes through embedding them into larger spaces–a technique that appears to be novel here–as well as on occasion without embedding.

We begin by introducing our new tool of classical promotion operators to generate hybrid codes and their associated codeword spaces. We provide some conditions that a stabilizer code must satisfy in order to be decodeable using our methods. Following this,

we extend our method of classical promotion operators to multiple operators, allowing for a greater amount of classical information to be conveyed.

3.1 Motivating Intuition

For stabilizer codes, we often require the states of interest to be the +1 eigenstates of the stabilizer generators. The proposed scheme here proposes taking advantage of the fact that, in principle, +1 being an indicator of no error could instead be taken to be any power of ω. This means that by simply applying a global shift of the syndrome values (and ensuring this cannot then be confused with a correctable error) will allow for a classical q-ary bit to be conveyed.

At this point, we define what hybrid codes are and in a more mathematical language the above observation and proposal.

Definition 41. *A hybrid code is a collection of quantum stabilizer codes $C_1, C_2, ...C_c$ such that each code corresponds to a different set of codewords, and thus we can associate each word space with a different classical result. These codes are denoted: $[[n, n-k : \log_q c, d]]_q$, where d is the quantum code's distance and we have assumed that these codes have the same number of stabilizers and distances [13].*

In order to be able to distinguish these codes from each other and not as errors within one code space, they must have distances between them of at least d. This means that to transform from one code space to the next the minimal weight of such a Pauli operator performing this action is at least d.

We consider a particular kind of qudit hybrid code utilizing the eigenvalues of the stabilizers. Essentially, we take advantage of relabelling the +1 portions of the syndrome table in order to pack in some classical information. This is proven by the rest of this chapter. Let $\mathbf{S}$ be a stabilizer code with generators s_i and parameters $[[n, n-k, d]]_q$. Let the corresponding codewords be given by $|c_w\rangle_0$, where the 0 subscript is just foreshadowing and the w value specifies the codeword within that code and has q^{n-k} values. By definition, we have:

$$s_i|c_w\rangle_0 = |c_w\rangle_0 \tag{3.1}$$

We now investigate codewords $|c_w\rangle_j$ such that:

$$s_i|c_w\rangle_j = \omega^{-j}|c_w\rangle_j \tag{3.2}$$

These $|c_w\rangle_j$ codewords are equivalently the states stabilized by the stabilizer code where s_i is replaced by $\omega^j s_i$. However, a priori, we have no reason to believe that these codewords ought to exist nor whether any method exists to convert from one to the next. Our first goal in this piece is to prove these facts.

3.2 Classical Promotion Operators

In this section we define our tool of *classical promotion operators*, show its operation, and provide a constructive proof of useful existence for a certain subset of stabilizer codes.

Definition 42 (Classical Promotion Operator)**.** *Let f be a qudit Pauli operator such that:*

$$f|c_w\rangle_j = |c_w\rangle_{j+1} \tag{3.3}$$

We call this f the Classical Promotion Operator.

Lemma 43. *We only need a single f in order to produce all code word sets.*

Proof. Since $f|c_w\rangle_j = |c_w\rangle_{j+1}$ this operator promotes by only a single word space at a time, and f is a qudit operator, we will scan through the entire space of codewords using this single operator by taking higher powers of f, up to $f^q = I$. □

Lemma 44. *Let f be defined as above and $s \in S^*$ be some pre-decided stabilizer generator for our code, where S^* is a set of independent stabilizer generators. Then the following must be true for all codewords within the same stabilizer code: $\omega sf = fs$.*

Proof. WLOG we assume we're doing the 0 to 1 promotion on some codeword c. Then:

$$f|c\rangle_0 = |c\rangle_1 \Leftrightarrow fs|c\rangle_0 = |c\rangle_1 \Leftrightarrow fs|c\rangle_0 = \omega s|c\rangle_1 \tag{3.4}$$

Noting that $\omega s|c\rangle_1 = \omega sf|c\rangle_0$, we have our result. □

Lemma 45. *Let s_i be a set of independent stabilizer generators which satisfy the following constraints for f:*

$$\phi(s_i) \odot \phi(f) = 1 \tag{3.5}$$

where $\odot$ is the symplectic product, defined by:

$$\phi(s_i) \odot \phi(f) = \oplus_k [\phi_z(f)_k \cdot \phi_x(s_i)_k - \phi_x(f)_k \cdot \phi_z(s_i)_k] \tag{3.6}$$

where $\cdot$ is standard integer multiplication mod q. This is equivalent to the condition specified in Lemma 44. These are the only conditions that f needs to satisfy, and these fully specify f.

Proof. Let $s = s_i \circ s_j$, where s_i and s_j satisfy the given condition. Since ϕ is a homomorphic map, this gives:

$$\phi(s) \odot \phi(f) = (\phi(s_i) \oplus \phi(s_j)) \odot \phi(f) = \phi(s_i) \odot \phi(f) \oplus \phi(s_j) \odot \phi(f) \tag{3.7}$$

which can easily be extended to larger compositions via induction. This proves that all higher order multiplications by stabilizers are simply addition mod q, as needed, thus all those constraints are satisfied if the first order ones are. We can undo the mapping ϕ up to the scalar coefficient, and so this result is also true for n-qudit operators. □

At this point we verify that the space of correctable errors is left unchanged by our classical promotion operators.

Lemma 46. *Syndrome tables are additive, and so pre-agreed upon applications of a Pauli preserves the space of correctable errors.*

Proof. The syndromes of two errors are additive mod the dimension as can be readily seen from the ϕ_∞ representation, and so applying an intentional, known Pauli will not alter the space of correctable errors, it will just apply a syndrome shift in all the syndrome tables. □

Corollary 47. *These classical promotion operators, which are Pauli operators, preserve the space of correctable errors.*

As of this point, we have merely stated properties that a classical promotion operator f must satisfy. We now provide an explicit construction for these given a stabilizer group $\mathbf{S}$. The following is a constructive proof of existence for one such operator that will perform our desired change.

Theorem 48. *For any stabilizer code with a non-single codeword space, there exists such an f operator for a special set of generators S^* chosen from the total stabilizer group.*

Proof. We begin by specifying S^*. We write the entire stabilizer group in the ϕ representation. Each row in the matrix A generated this way corresponds to one member in the stabilizer group. Let k be the number of stabilizer generators, then the matrix A has rank k. We may put this matrix into reduced row echelon form (RREF), where exactly k rows start with a 1 (this can always be done since for any nonzero number in $\mathbb{Z}_q$, there exists a unique multiplicative inverse) and those columns have a single 1. We define $S^* = RREF(A)$. This procedure also carries over to n-qudit operators by undoing our

map ϕ, up to the leading scalar, so we know that we have a collection of stabilizers still to refer to.

Now we show that we may construct an f that satisfies our conditions for the generators in S^*. We call these generators s_i as before. Then we need f to satisfy the system of equations:

$$\begin{bmatrix}\phi_z(s_i) & \phi_x(s_i)\end{bmatrix}\begin{bmatrix}-\phi_x(f)\\ \phi_z(f)\end{bmatrix} = \begin{bmatrix}\vec{1}\end{bmatrix}$$

or equivalently:

$$\phi(s_i) \odot \phi(f) = 1, \quad 1 \leq i \leq k \tag{3.8}$$

Since we know that there are k columns with a single 1, we may simply select those entries as being 1 in the f column vector. This gives us the values of $(-f_x|f_z)$, which is easily changed to $f = (f_x|f_z)$. This completes the proof, since we may invert the ϕ map to generate the n-qudit Pauli operator f. □

This proof shows that we may construct classical promotion operators for any stabilizer group, assuming both parties know the specially chosen generators for the code. We note that this is not the only way to construct the classical promotion operator f, but is a way to construct one such operator. Although we have shown that these classical promotion operators can exist, we have yet to show that they produce a decodeable encoding of classical information. In order for this to be the case, we must know that they cannot be mistaken for a correctable error.

Example 49. *This example illustrates the dependence of the minimal weight of f on the choice of S^*.*

First, we apply the method from the prior chapter to turn the Steane code into an invariant form. Recall from earlier that an invariant form for the Steane code, Ξ, is given by:

$$\phi_\infty(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1&0&0&0&0&0&1&0&0&0&0&1&1&0\\ 0&1&0&0&0&0&0&0&0&0&1&1&1&0\\ 0&0&1&0&0&0&1&0&0&0&1&0&1&0\\ 0&0&0&1&0&0&0&-1&1&0&0&0&0&1\\ 0&0&0&0&1&0&0&0&1&-1&0&0&0&1\\ 0&0&0&0&0&1&0&1&1&1&0&0&0&0 \end{array}\right]$$

Based on this, we may select our classical promotion operator to be $f^{-1} = ZZZZZZI$, with the generators S^* being these generators. This would allow for a classical bit to be communicated. However, we note that the error $IXIIIXI$ also generates the same

syndrome signature. This means that the action of f also appears in the space of errors we can correct, so this is not a good choice for S^*.

Suppose instead we pick as our invariant code the code generated by performing $r2 \leftarrow r2 - r1$, $r5 \leftarrow r5 - r4$:

$$\phi_\infty(\Xi) = \left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

Then $f = X \otimes Z^{-2} \otimes IIII \otimes X^2 Z^{-1}$ is a valid classical promotion operator. As can be verified, no weight 2 Paulis can generate the all 1 vector and so $wt(f) = d$, however, there exists errors which differ by the all 1 vector, so we cannot distinguish these errors with the inclusion of f.

Now we ask and answer the question of when we can ensure that the minimal weight Pauli that performs this action is at least the distance of the code *and* we can ensure that no set of errors differ by the all 1 vector.

Definition 50. *A representation of a stabilizer code is a particular set of priorly agreed upon set of generators for the stabilizer group.*

Since we have selected the all 1 vector, it would suffice to ensure that the support of the correctable errors is smaller than all k stabilizers.

Definition 51. *A representation of a stabilizer code S^* is said to have column weight $\mathcal{W}$ if:*

$$\max_i wt_i((S^*)^T) = \mathcal{W} \tag{3.9}$$

where wt is the number of nonzero entries in row i.

Lemma 52. *If the code S has a representation with column weight $\mathcal{W} < \frac{k}{2d}$, then:*

1. *The minimal weight of a classical promotion operator f for this code is $\geq \frac{k}{2\mathcal{W}}$*
2. *No errors in the space of correctable errors differ by a nonzero multiple of the all 1 vector*

Proof. First we show that the weight of f is at least $\frac{k}{2\mathcal{W}}$. Since each column only spans at most $\mathcal{W}$ syndrome bits, in order to span all k syndromes we need at least $\frac{k}{\mathcal{W}}$ nonzero entries, or a weight $\frac{k}{2\mathcal{W}}$ Pauli.

Second, a weight d Pauli can at most span $2d\mathcal{W}$ syndrome bits, thus as long as $\mathcal{W} < \frac{k}{2d}$ we cannot span the all 1 vector and so no errors can differ by this vector. □

Remark 53. *We use a hard distance d for the code (whereby any errors outside this radius, even if they're correctable by the original code, are treated as uncorrectable errors). If we do not use a hard distance then we cannot guarantee that we will not mistake these classical promotion operators with an error.*

With this, we can now transmit a single q-ary number for codes which satisfy this weight condition:

Corollary 54. *Let* $\mathbf{S}$ *be a stabilizer group for a stabilizer code with* $[[n, n-k, d]]_q$*, with* q *prime, then using the scheme described above, we may create a* $[[n, n-k : 1, d]]_q$ *hybrid code encoding a number in* $1 \to q$*.*

Proof. This follows since we may encode any of $1 \to q$ and our scheme preserves the code, and thus preserves the number of logical qudits and the distance of the code. Lastly, since f has weight d this ensures that each code is sufficiently separated that errors cannot mix the code spaces. □

The implication of this is that in a q level qudit system, we can algorithmically turn some stabilizer codes into q codes and thus encode any number from $1 \to q$, or equivalently $\log_2 q$ bits. This procedure also produces all the other space's codewords, which means knowing a single space's codewords generates all the rest. Therefore, knowing a single stabilizer code and its codewords is sufficient for this procedure, which removes the sometimes onerous task of discovering these for each code in the hybrid code.

3.3 Extension to Multiple Classical Promotion Operators

In the prior section we showed how to include a single q-ary bit, however, we would like to transmit more than a single number. The intuition for the first part is that causing a total shift to all syndrome values will still leave a correctable error and allow for the receiver

to know which classical bit was being conveyed by simply taking a majority vote in the syndromes.

We now introduce the idea of factoring these classical promotion operators in order to increase the number of bits transmitted. This idea follows the logic that if only a small fraction of the syndromes are needed to determine which error occurred, then we may apply a shift to the syndrome table on blocks and still be able to faithfully determine which classical promotion operators have been applied.

Definition 55. *The factors of our Classical Promotion Operators are expressed as:*

$$f = \prod_{s \in \eta} f_s \tag{3.10}$$

where η is a partition of the stabilizer generators' indices.

We apply each of these f_s, and chosen power, to indicate a particular value. We can trivially achieve this by taking only those elements in the partitions as the support of the factors. Each of these factors f_s will only not commute with those stabilizers in its support. Even still, this can be used to create a bijective effect to encode and decode classical information.

Before, we required that less than half the stabilizers are needed in order to determine the error incurred. In order for this to work, we need to make the column weight bound even smaller for the code. Following the same arguments as above, this provides the tighter bound of $\mathcal{W} < \frac{k}{2dc}$.

Remark 56. *The generators of the stabilizer must be known to both parties ahead of time as well as the decomposition of the Classical Promotion Operator into its factors.*

Remark 57. *Given the way that this code is constructed, we can easily concatenate stabilizer codes to encode more classical q-ary numbers, but the classical rate per block is still fixed.*

3.3.1 A Simple Family of Low Column Weight Codes

Although some low column weight codes may exist, we provide a way to generate our own. Let Ξ be a stabilizer code with parameters $[[n, n-k, d]]_q$. Then the code generated by a $d+1$-fold repetition of Ξ will have column weight $\leq k$ out of $(d+1)k$ total stabilizers for this concatenated code. The distance of this code is still d, so we have traded off being

able to correct d errors per block for only being able to correct d errors across the entire repetition. We can extend this to multiple classical bits by simply taking a $c(d+1)$-fold repetition of the code Ξ.

This then means that we may transform most stabilizer codes into hybrid codes with this method, although perhaps not with as excellent parameters. Performing this concatenation sacrifices some quality, so instead codes which naturally have low column weight would be better candidates for this scheme.

3.4 Conclusion and Discussion

In this chapter we have introduced a way to transmit classical information using stabilizer codes of low column weight. We note, however, that codes of this form likely have inferior parameters to other code choices and so the transmission of this classical information comes at the cost of choosing a lower quality code. Although one cannot achieve both high distance and a perfect stabilizer code, this particular restriction likely reduces to such a large loss that we would be better off just sacrificing an encoded qudit to transmit the information instead. This then indicates that degenerate stabilizer codes are strong candidates for hybrid codes, however, this is left as a future direction– or alternatively showing that degenerate codes are themselves also worse than nondegenerate codes in notable regards could suffice for showing that hybrid stabilizer codes are not generally efficient.

Chapter 4

Conclusion and Future Directions

Many tools developed for classical error-correcting codes have been imported into the quantum error-correction language. However, this does not encapsulate all the results in quantum error-correction. In this work we have focused on some features that are possible when working with qudit stabilizer codes. In particular, we have shown that we may embed codes into larger spaces without compromising parameters, at least for infinitely many primes, and that inscribing codes can cause a reduction in the quality of the codes. This is a somewhat surprising result, but it has applications to generating codes for qudit systems. Beyond that, we proceeded to show a scheme that, under certain conditions, would allow for classical information to be transmitted with quantum information and transform many families of well-known stabilizer codes into hybrid codes, another seemingly new result. Prior to this, only the extremes of purely theoretical and explicit examples have been shown, this strikes the middle by providing many examples all at once and perhaps inspiring further extensions.

Despite what has been shown, there are still possibly important extensions to what has been done. Under which conditions might qudit codes outperform those of qubit codes, and when they do outperform, is it a significant improvement or rather modest? Qudit systems are usually harder to control with precision than qubit systems, so will these improvements be able to compensate for associated error rate increases?

Besides this, we hope that some of the results of this thesis can be applied in experimental settings, providing protection to qudit systems that otherwise would have been arduous to find or providing a method of conveying classical information without cost. Additional applications of these results are not known at this time, but these results can be added to the repertoire of tools available to try to correct errors in quantum computing systems.

References

[1] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *arXiv preprint quant-ph/9906129*, 1999.

[2] Erdal Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on information Theory*, 55(7):3051–3073, 2009.

[3] Claude Berrou, Alain Glavieux, and Punya Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Proceedings of ICC'93-IEEE International Conference on Communications*, volume 2, pages 1064–1070. IEEE, 1993.

[4] Raj Chandra Bose and Dwijendra K Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and control*, 3(1):68–79, 1960.

[5] A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[6] Xie Chen, Bei Zeng, and Isaac L Chuang. Nonbinary codeword-stabilized quantum codes. *Physical Review A*, 78(6):062315, 2008.

[7] Richard Cleve, Daniel Gottesman, and Hoi-Kwong Lo. How to share a quantum secret. *Physical Review Letters*, 83(3):648, 1999.

[8] Igor Devetak and Peter W Shor. The capacity of a quantum channel for simultaneous transmission of classical and quantum information. *Communications in Mathematical Physics*, 256(2):287–303, 2005.

[9] Daniel Gottesman. Stabilizer codes and quantum error correction. *arXiv preprint quant-ph/9705052*, 1997.

[10] Daniel Gottesman. Fault-tolerant quantum computation with higher-dimensional systems. In *NASA International Conference on Quantum Computing and Quantum Communications*, pages 302–313. Springer, 1998.

[11] Daniel Gottesman and Isaac L Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402(6760):390, 1999.

[12] Markus Grassl, Thomas Beth, and Martin Roetteler. On optimal quantum codes. *arXiv e-prints*, pages quant–ph/0312164, Dec 2003.

[13] Markus Grassl, Sirui Lu, and Bei Zeng. Codes for simultaneous transmission of quantum and classical information. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1718–1722. IEEE, 2017.

[14] Zhuo Li and Lijuan Xing. No more perfect codes: Classification of perfect quantum codes. *arXiv preprint arXiv:0907.0049*, 2009.

[15] David Poulin, Jean-Pierre Tillich, and Harold Ollivier. Quantum serial turbo codes. *IEEE Transactions on Information Theory*, 55(6):2776–2798, 2009.

[16] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[17] Peter Shor and Raymond Laflamme. Quantum analog of the macwilliams identities for classical coding theory. *Physical review letters*, 78(8):1600, 1997.

[18] Andrew Steane. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452(1954):2551–2577, 1996.

[19] Andrew M Steane. Error correcting codes in quantum theory. *Physical Review Letters*, 77(5):793, 1996.

[20] Mark M Wilde, Min-Hsiu Hsieh, and Zunaira Babar. Entanglement-assisted quantum turbo codes. *IEEE Transactions on Information Theory*, 60(2):1203–1222, 2013.

[21] Mark M Wilde and Joseph M Renes. Quantum polar codes for arbitrary channels. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 334–338. IEEE, 2012.